Real-Time Identification of Cyber-Physical Attacks on Water Distribution Systems via Machine Learning Based Anomaly Detection Techniques

Article *in* Journal of Water Resources Planning and Management · July 2018

DOI: 10.1061/(ASCE)WR.1943-5452.0001023

CITATIONS
61

READS
1,149

4 authors, including:

Ahmed Abokifa
University of Illinois at Chicago
46 PUBLICATIONS **861** CITATIONS

SEE PROFILE

Pratim Biswas
Washington University in St. Louis
530 PUBLICATIONS **26,534** CITATIONS

SEE PROFILE

Accepted for publication in the ASCE Journal of Water Resources Planning and Management – July 2018

Real-Time Identification of Cyber-Physical Attacks on Water Distribution Systems via Machine Learning Based Anomaly Detection Techniques

Ahmed A. Abokifa[1], Kelsey Haddad[1], Cynthia Lo[1], and Pratim Biswas[1]

[1]Washington University in St. Louis

Abstract. Smart water infrastructures are prone to cyber-physical attacks that can disrupt their operations or damage their assets. An algorithm is developed to identify suspicious behaviors in the different cyber-physical components of a smart water distribution system. The algorithm incorporates multiple modules of anomaly-detection techniques to recognize different types of anomalies in the real-time monitoring and control data. Trained artificial neural networks are used to detect unusual patterns that do not conform to normal operational behavior. Principal component analysis is conducted to decompose the high-dimensional space occupied by the sensory data to uncover global anomalies. The algorithm is trained using a historical dataset of trusted observations and tested against a validation and a test dataset, both featuring a group of simulated attack scenarios. The proposed approach successfully identifies all the attacks featured in the BATtle of the Attack Detection ALgorithms (BATADAL) datasets with high sensitivity and specificity. Nevertheless, the performance is sensitive to high background noise in the sensory data.

Keywords: water distribution systems; cybersecurity; anomaly detection; machine learning

1. Introduction

Numerous water utilities have recently started adopting smart technologies in their drinking water distribution systems (DWDSs) to improve their overall performance, efficiency, and reliability. Smart water distribution networks belong to the group of modern cyber-physical systems (CPSs), in which on-line monitoring, data collection and transmission, real-time computation, and automated operation of the functional processes are tightly integrated (Lee 2008). Hence, smart water grids commonly rely on a coordinated network of distributed sensors and remote actuators, which are typically linked to programmable logic controllers (PLCs) managed in real time by a supervisory control and data acquisition (SCADA) system (Shamir and Salomons 2008). The increased interest in embracing smart network technologies over the past decade has been complemented with a consistent growth in the development of related industrial tools and solutions, such as advanced metering technologies, sensor networks, data analytics tools, and automation systems. Nevertheless, the enhanced connectivity instigated by such advanced control technology has concurrently opened the door to a novel class of security vulnerabilities that were not inherent to the physical infrastructure system (Laszka et al. 2017; Rasekh et al. 2016).

Despite the numerous merits of implementing modern networking technologies in the sector of critical infrastructure systems, linking the physical components of the infrastructure with cyber-space can expose these systems to the vast realm of cyber-based threats. From a national security standpoint, water infrastructure systems, including drinking water treatment facilities and distribution networks, possess a sensitive disposition given the critical role they play in the sustainable development of modern communities. This role makes WDSs a highly attractive target for cyber-attacks that can be potentially perpetrated by terrorists, subversives, and adversary states. These attacks can target the SCADA module, the sensors that monitor the system's processes, the PLCs that locally operate the physical components of the infrastructure or the wireless communication routes between the different elements of the CPS. Such cyber-based attacks are capable of remotely perturbing the performance of the system, providing unauthorized parties with access to critical and confidential information, and -if sophisticated enough- can result in physical damage to the assets of the infrastructure. Additionally, such attacks can compromise the water quality by altering automated treatment schemes or by targeting water quality sensors to suppress contamination warnings, which can pose a significant threat to public safety. The last decade has witnessed a spike in the number of cyber-security incidents involving water infrastructure systems. In 2015, the US Department of Homeland Security (DHS) reported that the Industrial Control Systems-Cyber Emergency Response Team (ICS-CERT) received and responded to 25 cyber-related incidents that targeted water and wastewater systems, making it the third highest targeted sector by cyber incidents after critical manufacturing and energy (DHS ICS-CERT 2015). Two well-cited examples for the type of threats that modern water systems face are the cyber-attacks that targeted the SCADA systems of the Maroochy Water Services in Queensland, Australia (Slay and Miller 2008), and the water utility of Boca Raton, Florida (Horta 2007).

To mitigate similar threats in the future, previous studies discussed the importance of establishing a mature cyber-security culture within the water industry in order to reduce the susceptibility of smart WDSs to cyber-attacks (Panguluri et al. 2017). Moreover, imposing additional security measures on the different components of the CPS, including the remote sensors/actuators, the communications network, and the SCADA module (Mathur 2017) can potentially enhance their resilience in the face of cyber-attacks. However, the relatively extended periods of the WDS operation implies that the probability that one of its components is attacked at least once during its lifetime is non-negligible (Taormina et al. 2016). Thus, developing strategies that can effectively detect any abnormal behavior in the different domains of the cyber-physical system in real-time is of prime importance in order to prevent service interruptions and protect both the system's assets and public health. The general problem of the detection and identification of attacks on cyber-physical systems, as well as intrusion detection for the underlying SCADA systems, has been addressed by several studies (Gao et al. 2010; Maglaras and Jiang 2014; Pasqualetti et al. 2013). Nevertheless, most of the previous efforts were devoted to detecting attacks on smart power grids and communication networks (Kosut et al. 2010;

Sridhar and Govindarasu 2014), with disproportionately less emphasis on water infrastructure systems.

For drinking water distribution networks, the broad topic of detecting physical-based threats has been previously investigated by several studies that mainly focused on the classical problems of fault detection (Eliades and Polycarpou 2012; Izquierdo et al. 2007; Srirangarajan et al. 2013) and contamination event detection (Arad et al. 2013; Housh and Ohar 2017a; b; Ohar et al. 2015; Perelman et al. 2012). While the problems of detecting cyber-based and physical-based threats both belong to the general class of event identification, three important differences exist between these two problems, namely the spatial and temporal resolutions, and attack concealment (Housh and Ohar 2018). The most fundamental difference is that cyber-based attacks can be potentially concealed by the attacker in order to cover any traces in the sensory data. Attack concealment can be done by means of a deception attack, in which the attacker alters the observations received by the SCADA system by sending false plausible values instead of the real suspicious ones (Taormina et al. 2017). One possible way to do this is by conducting a "replay attack" in which observations during normal operations are recorded by the attacker, and then replayed to the SCADA system during the attack, which makes the detection of such stealthy attacks a very challenging task.

A few attempts have been made to tackle the problem of cyber-physical attacks detection on water infrastructure systems. Amin et al. (2013a; b) investigated the detection and isolation of cyber-attacks on the SCADA system of an irrigation canal network using an approximate hydrodynamic model. Yet, their work was not extended to pressurized looped WDSs. Almalawi et al. (2016) proposed an intrusion detection method to detect SCADA tailored attacks based on a data-driven clustering technique with a demonstration on a simple WDS model example. Recently, Taormina et al. (2017) developed a modeling framework to assess the hydraulic response of water distribution networks to cyber-physical attacks. Nevertheless, their work focused on simulating and characterizing cyber-attacks rather than their detection and identification.

This study aims to develop an approach for the identification of cyber-physical attacks (CPAs) on WDSs in real-time by detecting suspicious anomalies in the SCADA observations. We first establish the different types of anomalies that can be inflicted by CPAs. We then demonstrate the use of different machine-learning techniques, namely principal component analysis (PCA) and artificial neural networks (ANNs), for the detection of the various classes of anomalies, and their integration for the identification of CPAs. The design goals of the detection algorithm are: i) to determine the existence of an ongoing attack with maximum speed and reliability; ii) to avoid issuing false alarms and to recognize when the system is no longer under attack, iii) to identify which components of the cyber-physical infrastructure have been compromised during the attack, and iv) to reliably distinguish between anomalies caused by cyber-attacks and measurement noise. The algorithm is first trained using a trusted set of

SCADA observations, then validated, and tested against two different datasets comprising a group of simulated malicious attack scenarios.

2. Case Study and Datasets

A medium-sized water distribution network, C-Town network, is used as a benchmark for algorithm development and application. C-Town network, displayed in Figure S1, consists of 388 nodes connected by 429 links. Water storage and distribution across the network are supplied by seven tanks (T1-T7) whose water levels control the operation of eleven pumps (PU1-PU11) grouped into five pumping stations (PS1-PS5), and one control valve (V2). Additional network details can be found in the EPANET input file "CTOWN.inp" in the supplementary material. This network was recently used for the BATtle of the Attack Detection ALgorithms (BATADAL) (Abokifa et al. 2017; Taormina et al. 2018) (https://batadal.net/). The system implements a smart water grid technology featuring a set of remote sensors and actuators in order to monitor and control the operation of all tanks, functioning valves, and pumping stations. These sensors/actuators are connected to nine PLCs that transmit the data to a centralized SCADA system, which coordinates the network operation in real-time (Figure S1).

As described in the BATADAL, the system is subject to a group of simulated cyber-physical attacks that perturb the functionality of the actuators, alter the readings of the deployed sensors, and interfere with the connections between the networked components in the cyber-layer. These attacks are generated with the *MATLAB* modeling toolbox *epanetCPA*, which allows the simulation of the hydraulic response of water distribution systems to cyber-physical attacks using EPANET (Taormina et al. 2017). Three independent datasets are used to *train*, *validate*, and *test* the algorithm, respectively. The first (training) dataset comprises historical SCADA observations generated for a period of one year (on an hourly basis) prior to the deployment of the smart technology. This dataset is guaranteed to contain no attacks, and is hence used to train the algorithm to recognize the normal behavior of the system. The second (validation) dataset was generated for a six-month period following the deployment of the smart technology and contains seven different simulated attack scenarios (Table S2). This dataset is used to validate the performance of the proposed technique in detecting cyber-physical attacks. In addition, a sensitivity analysis is performed at this step to adjust the different parameters of the algorithm to yield best detection performance. Similarly, the third (test) dataset was generated for a three-month period and contains seven simulated attacks (Table S3). The test dataset is used herein to verify the efficiency of the tuned algorithm in identifying multiple attacks in a real-time stream of observations that the algorithm has never seen before (i.e. was not part of the training/validation process). The three datasets can be found in comma-separated values (.csv) files in the supplementary materials together with a detailed description of their contents in Section S1.

3. Anomaly Characterization

In an ideal scenario, the utility should possess an accurate hydraulic model of its WDS that is routinely calibrated by the collected monitoring data. The presence of such a model would be of

value in detecting abnormal system behaviors and can provide useful insights if used in conjunction with an algorithm that analyzes the data from the SCADA system. In such a framework, detection of cyber-physical attacks (CPAs) is driven by precise simulations of the system dynamics (Housh and Ohar 2018). Nevertheless, a clear limitation arises for such *"model-based"* approaches in cases where accurate water demand patterns in the system are not readily available or if the network model is computationally expensive prohibiting real-time evaluations (Abokifa et al. 2016). Thus, to reduce the dependence of the designed algorithm on the availability of a calibrated network model, the attack detection problem considered herein is exclusively dependent on the given SCADA observations. Hence, identification of attacks is only achieved through employing anomaly detection techniques that target discovering irregularities in the data induced by cyber-physical attacks. In other words, the main aim of the proposed "*data-based*" approach is to search for *extended inconsistencies* in the observations that can be interpreted as indicators or "fingerprints" inflicted by an ongoing attack. An anomaly can be generally defined as a data point (or a series of points) that does not conform to a well-defined notion of normal behavior (Chandola et al. 2009; Hawkins 1980). Hence, a first step to detect anomalies in a given set of observations is to define a domain (subset) of observations that are trusted to represent the normal behavior of the system. In our case, this domain is defined by the first (training) dataset of historical SCADA observations. Taormina et al. (2017) provides a thorough discussion on how the operation of WDSs is altered in response to CPAs. Herein, we discuss the different classes of anomalies that can be induced by CPAs in the sensory data of WDSs, and propose a different approach to detect each class.

3.1. Simple Outliers vs Contextual anomalies

In the context of detecting cyber-physical attacks on WDSs, the definition of an anomaly should not be limited to the identification of "simple outliers", i.e. data points that lie beyond certain historical fences defined based on a reference dataset representing normal (expected) behavior. An example of this is an attack that persistently shuts down a pumping station resulting in excessively low levels in one or more tanks. However, a cyber-physical attack can interfere with the performance of one of the infrastructure components in a way that alters its operational patterns -compared to the normal conditions-, while maintaining its performance characteristics (e.g. tank level or pumping flow rate) within the normal historic min/max bounds. In this case, the anomalous pattern is described as a "contextual anomaly" rather than a simple outlier, which means that the suspicious observation is anomalous within a specific temporal context based on the previous observations, regardless of its magnitude.

To illustrate the difference between these two types of anomalies, Figure 1-a shows a series of anomalous data points observed in the flow data of Pump 1 in the second (validation) dataset. These anomalies correspond to the third attack event. As can be seen, the suspicious data points are noticeably lower in magnitude compared to the minimum historic bound recorded in the first (training) dataset. On the other hand, Figure 1-b shows an anomalous pattern in the flow data of Pump 7, which corresponds to the fifth attack in the validation dataset. In this case, the

magnitudes of the anomalous points are well within the previously defined historic bounds. Nevertheless, pump performance has been clearly interrupted as shown by the evident change in the flow patterns.

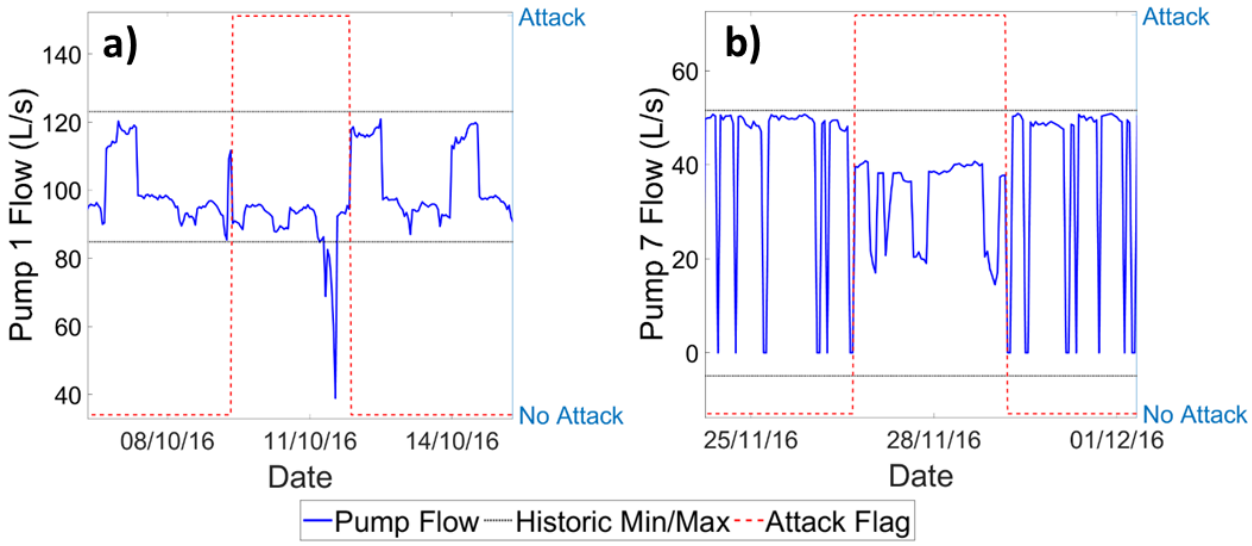

Figure 1. Validation dataset observations for: (a) Pump 1 flow during attack 3 (simple outlier), and (b) Pump 7 flow during attack 5 (contextual anomaly). Attack Flag indicates when the system is subject to one of the simulated attacks.

In this study, "simple outliers" are detected by comparing the observations to certain statistical fences determined by a given number of standard deviations below and above the mean, and interquartile ranges above and below the upper and lower quartiles. On the other hand, identifying "contextual anomalies" that do not conform to the regular operational patterns requires training the algorithm to "learn" these patterns in the first place. This can be accomplished using a supervised machine-learning algorithm that is trained to forecast a future data point from a series of previous observations. In this study, Artificial Neural Networks (ANNs) are used for this task due to their known capability of modeling complex nonlinear relationships. ANNs were previously used to forecast dynamic time series patterns of water resources variables (Maier and Dandy 2000), and applied to forecast the dynamic hydraulic and water quality states for water distribution networks by creating surrogate models (meta-models) (Broad et al. 2010, 2015; May et al. 2008; Razavi et al. 2012; Romano and Kapelan 2014). In this study, we use ANNs to model the patterns of each individual sensor/actuator, and anomalies are detected by comparing the observed data points against the values predicted by the ANNs model.

3.2. Stealthy attacks and global anomalies

As previously discussed, one of the most challenging aspects regarding the detection of cyber-based threats on water distribution systems, and a one that sets it apart from the traditional problem of physical fault detection, is the potential for attack concealment. This may involve replaying part of the historical observations to make the data appear normal. Such *stealthy*

attacks may interfere with the performance of multiple components of the WDS without significantly altering the individual characteristics of any of them. For instance, in the seventh attack scenario featured in the validation dataset, the attacker conceals the SCADA readings of the water level in T4, and the flow and status of pumps PU6/PU7 with a *replay* attack. Hence, detection methods that separately analyze the data from each of these individual sensors, such as statistical fences and ANN models discussed in the previous section, may miss such attacks since the induced anomalies in each of the data arrays will likely be below their detection limits. This highlights the need for incorporating an additional detection method that aims to discover such "global anomalies" by analyzing the combined data obtained from all sensors in the multi-dimensional space.

In this study, we use principal component analysis (PCA), a linear dimensionality reduction technique (Cunningham and Ghahramani 2014; Jolliffe and Cadima 2016), to project the observed multi-dimensional data from all sensors onto a set of principal components. By doing this, anomalies in the data arrays corresponding to all the monitors are synergized, which gives a magnified response enabling the capture of global anomalies corresponding to stealthy attacks (Lakhina et al. 2004). The fundamental concept here is that, for water distribution systems, a certain degree of correlation exists between the global observations made by all the monitors (sensors/actuators) at any given time step. When the attacker conceals the readings of a subset of these sensors during a certain time step, the concealed readings will not follow this correlation even if they still follow the normal pattern for the concealed component (more discussion in Section S2).

4. Algorithm Development and Anomaly Detection

The detection algorithm is designed as an "ensemble model" featuring four different modules. Each module targets the detection of a specific class of outliers/anomalies separately (Figure 2). The first module checks whether the given SCADA observations follow the actuator rules defined for the system, while the second module focuses on finding simple statistical outliers with excessively high or low magnitudes. The third module aims at revealing contextual anomalies using trained ANN models, and the fourth module targets discovering global anomalies in the multi-dimensional space using PCA. Anomalies from the four modules are then integrated by means of an alarm watch *window* method as discussed in the following subsections.

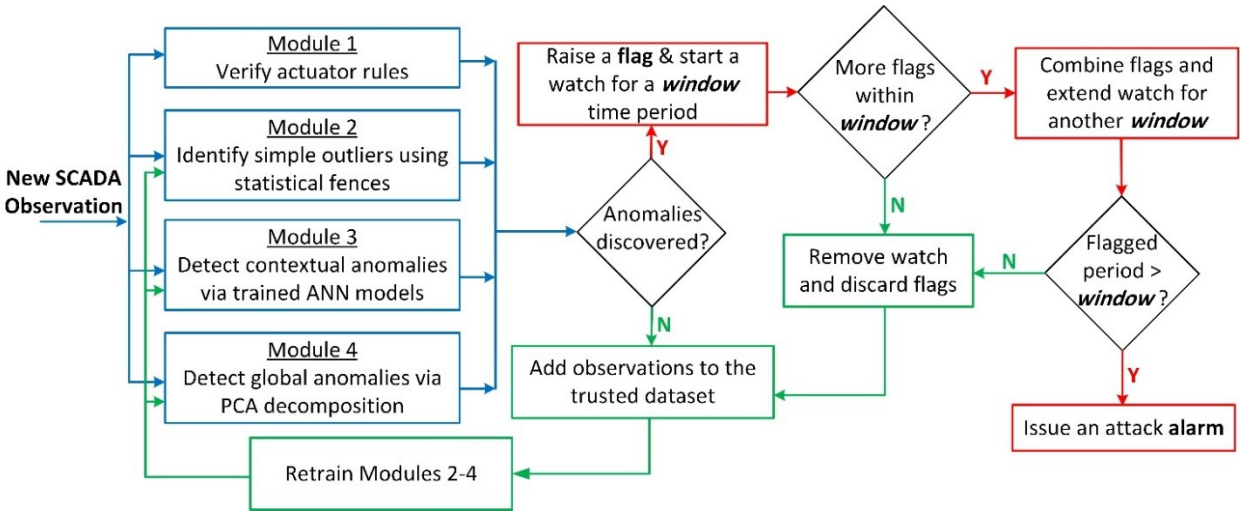

Figure 2. Schematic diagram of the attack detection algorithm

4.1. Module 1: verification of actuator rules

The first module checks that the operational statuses of the pumps and valves follow the appropriate control rules based on the observed water levels in their controlling tanks at all times. For example, in the given C-Town system, Pump 1 operation is controlled by the water level in Tank 1. Actuator control rules dictate that a drop in the tank level below 4 m triggers the operation of Pump 1 by switching it on, which should change its status to (S_PU1 = 1). Pump operation should carry on until the tank level goes above 6.3 meters, at which the pump should be switched off changing the status to (S_PU1 = 0). Thus, these rules might be violated if the system is subject to a "deception" attack that manipulates the communications between the PLC connected to the tank level sensor and that connected to pump actuator. The module returns a flag for each data point that violates any of the rules depicted in the input file.

4.2. Module 2: detection of simple outliers via statistical fences

The second module focuses on detecting outliers with suspiciously high/low magnitudes compared to statistical bounds based on historical observations. Let $\boldsymbol{A}$ be the training dataset, which is an $(N \times M)$ matrix comprising N observations (time-steps) from M monitors (sensors/actuators). Hence, $\boldsymbol{A} = [\boldsymbol{\xi_1}, \boldsymbol{\xi_1}, \dots, \boldsymbol{\xi_M}]$, where $\boldsymbol{\xi_i} = \left[x_{1,i}, x_{2,i}, \dots, x_{j,i}, \dots, x_{N.i}\right]^T$ is the vector of N observations made by monitor i, and $x_{j,i}$ is the observation made by monitor i during time step j. A simple statistical approach to detect outliers in a new observation made by one of the monitors can be done by specifying certain boundaries for each monitor i defined by multiples of the standard deviation above and below the mean, or based on the interquartile range (IQR) as follows:

$$\text{Upper fence: } u_i = \max(\mu_i + n_i^u \sigma_i \ , \ \ Q3_i + m_i^u [Q3_i - Q1_i]) \tag{1}$$

$$\text{Lower fence: } l_i = \min(\mu_i - n_i^l \sigma_i \ , \ \ Q1_i - m_i^l [Q3_i - Q1_i]) \tag{2}$$

where, u_i and l_i are the designated upper and lower boundaries for monitor i; μ_i and σ_i are the mean and standard deviation of data vector $\boldsymbol{\xi}_i$, respectively; $Q1_i$ and $Q3_i$ are the lower and upper quartiles of the data array $\boldsymbol{\xi}_i$, respectively; n_i^u and n_i^l are the standard deviation multipliers representing the upper and lower fences; and m_i^u and m_i^l are the interquartile range multipliers for the upper and lower fences, respectively. The upper and lower multipliers of the standard deviation and the IQR (i.e. $n_i^{u,l}$ and $m_i^{u,l}$) are defined based on the maximum and minimum values recorded in the training dataset for each monitor. The first module compares new observations by any of the monitors, e.g. in the validation or test datasets, to the upper and lower fences, and returns a flag for any observation lying beyond these fences, with an arbitrary tolerance margin (5%).

4.3. Module 3: detection of contextual anomalies via ANN models

Herein, ANN models are used to construct the forecasted patterns for each of the monitored hydraulic parameters based on the learned system performance from the training dataset to uncover anomalous patterns. To this end, ANN models are designed to predict a single future reading from a series of previous observations for each of the monitors. A multi-layer perceptron (MLP) neural network model is used, which is a computational model consisting of multiple layers of inter-connected artificial neurons. Each neuron performs a nonlinear computation, and the weighted sum of the outputs from all the neurons in one layer is fed to the neurons of the following layer (feed-forward NN).

4.3.1. ANN models architecture

A separate ANN model is constructed for each individual monitor. The output layer consists of one neuron, while the input layer consists of a set of I input neurons, which comprises the array of previous observations used to predict the next observation. A single layer of hidden neurons is employed, with a number of hidden neurons that is equal to the number of neurons in the input layer. Therefore, each monitor's training data array $\boldsymbol{\xi}_i$ containing N observations is split into a group of $(N - I)$ training sets, with each set consisting of I inputs and one desired output. For example, the first set for any monitor i will comprise observations $[x_{1,i}, x_{2,i}, \dots, x_{I,i}]$ as inputs and $[x_{I+1,i}]$ as the desired output. The output of each neuron k can be written as:

$$y_k = g\left(\sum_{r=1}^{I} x_{r,k} w_{r,k} + w_{0.k}\right) \tag{3}$$

where, y_k is the output of neuron k, $x_{r,k}$ and $w_{r,k}$ are the input and the weight coefficient of neuron k in the current layer from neuron r in the preceding layer, $w_{0.k}$ is the bias coefficient of neuron k, and g is the hyperbolic-tangent sigmoid transfer function defined as:

$$g(z) = \frac{2}{1 + e^{-2z}} - 1 \tag{4}$$

By tuning the weights assigned to each neuron, the desired relationship between the inputs and the outputs of the ANN can be established. This process is known as training or learning, and in this study, it is done using a backpropagation algorithm that adjusts the weight coefficients of the ANN through minimizing the error between model predictions and the observations for each one of the $(N - I)$ training sets using the Levenberg-Marquardt optimization algorithm incorporated in the MATLAB R2016a Neural Network Toolbox™.

The performance of the ANN forecasting model mainly depends on its architecture. Typically, increasing the number of neurons enhances the accuracy, yet it also increases the complexity and computational burden of training the model. Figure S3-c shows the maximum training error generated from the constructed ANN models for the water level in Tank 1 with different architectures. Each architecture is characterized by a different size (i.e. number of neurons) for the hidden layer, which is also equivalent to number of input observations I. Error bars represent the standard deviation of five different ANN models constructed for each architecture. As expected, the accuracy of the trained ANNs increases consistently with increasing the size of the input/hidden layer since the number of previous observations propagated into the model to forecast a future data point increases. Nevertheless, no significant enhancement is observed for architectures with $I > 40$ neurons/hrs. Thus, an input/hidden layer size of $I = 40$ neurons/hrs is selected for all ANN models to achieve an acceptable performance with minimal complexity.

4.3.2. Data Pre-processing

Prior to constructing the ANN models, we observed that the raw hydraulic data for tank levels, pumping flow rates, and pressures, comprises sharp fluctuations on short time intervals (Figure 1), which makes training the ANNs a complex and computationally demanding process. Therefore, instead of using raw observations, preprocessing is first conducted by projecting the data into a spectral frequency domain representation of the time series signal in order to generate a smoothed form of the data that preserves the same structure but with less frequent fluctuations. To do that, the Fast Fourier Transform (FFT) of the signal is obtained to decompose the time series data into its underlying frequencies for each of the given data arrays ($\boldsymbol{\xi_i}$). Then, the raw data is smoothed using a third degree low pass Butterworth filter, with a cutoff frequency that corresponds to 50% of the cumulative amplitude of the signal. Figure 3 shows the flow data for pumps 1 and 7 before and after smoothing for the same attack events displayed in Figure 1. It demonstrates that the filtered data clearly features the same anomalies existing in the raw data for both types of anomalies, but the high frequency fluctuations are significantly smoothed. In addition to facilitating the ANN training process, smoothing the data decreases the training error, which results in a higher forecasting accuracy for the trained models and enhances anomaly detection.

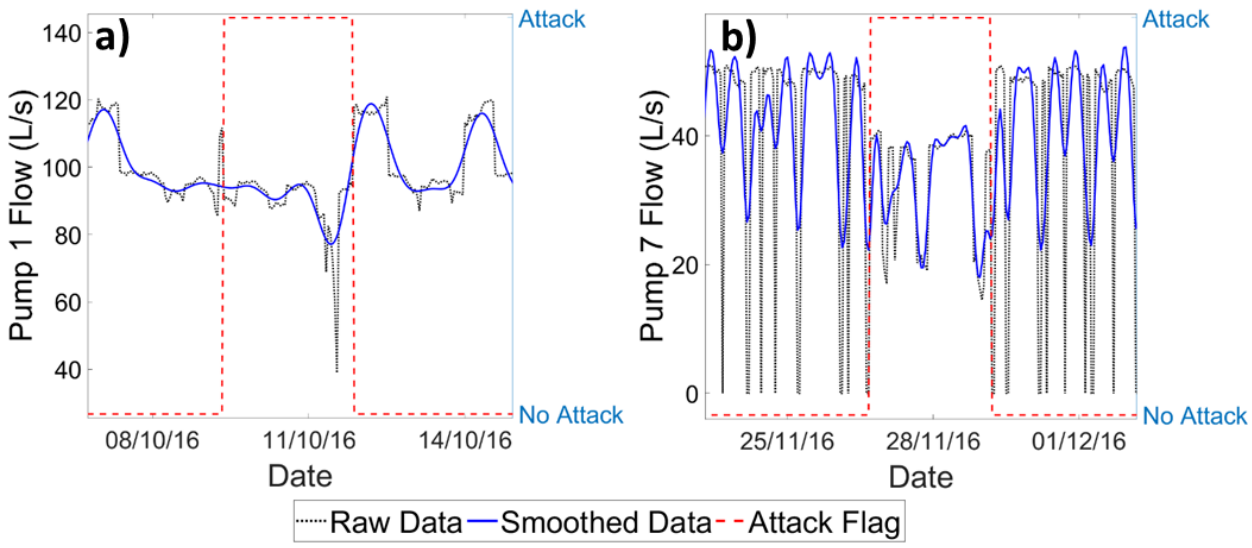

Figure 3. Raw vs. smoothed data for: (a) Pump 1 flow (simple outlier), and (b) Pump 7 flow (contextual anomaly).

4.3.3. Anomaly detection via ANN

To define anomalies using the trained ANNs, a different technique should be used in place of the previously described statistical fences (Equations 1, & 2). Herein, anomalous observations are defined based on their deviation from the values predicted by the ANN model, and thus, the accuracy of the model in forecasting normal observations should be considered in such evaluation. Therefore, outliers should be defined based on how large their prediction errors are in comparison with the maximum error encountered in forecasting observations of the training dataset. For each monitor i, the maximum training error can be written as:

$$\epsilon_i^{train} = \max_s\left(\left|o_{i,s}^{train} - p_{i,s}^{train}\right|\right) \quad (5)$$

where, $o_{i,s}$ and $p_{i,s}$ are the desired (observed) and predicted outputs of training set s for monitor i. In this study, observations in both the validation and test datasets that have a prediction error greater than the maximum training error multiplied by a specific factor α are considered anomalous:

$$\left|o_i^{valid/test} - p_i^{valid/test}\right| \geq \alpha\ \epsilon_i^{train} \quad (6)$$

To demonstrate this, Figure 4-a shows the smoothed flow observations for pump 10 for the validation dataset as well as the predicted values by the ANN model. The depicted threshold is equal to ($\pm\epsilon^{train}$). The deviation (error) between the predicted and observed points during the attack event is clearly larger than that when the system is not under attack (Figure 4-b).

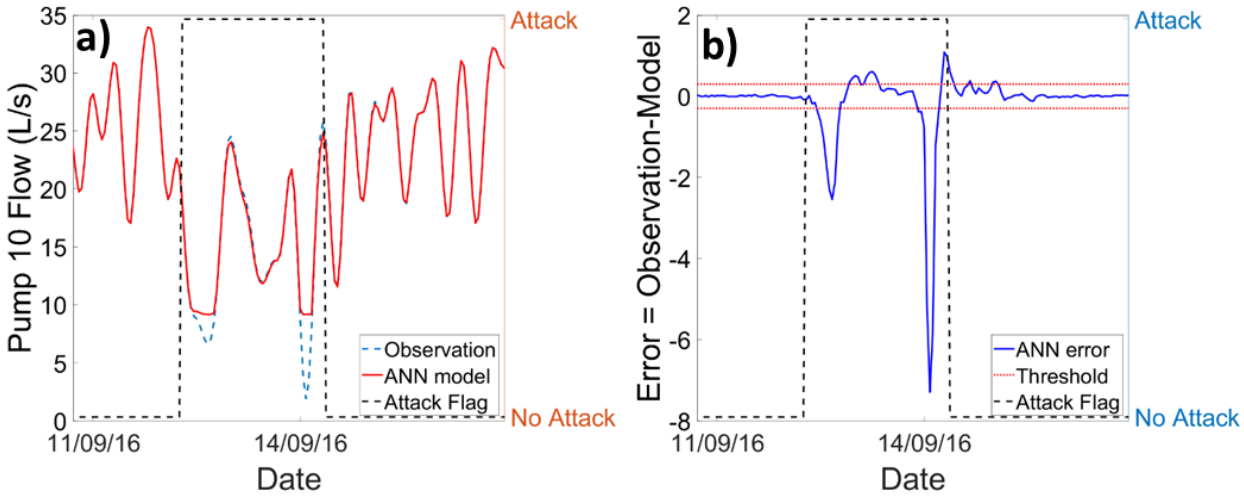

Figure 4. (a) Smoothed data for pump 10 flow, and (b) error between observed and modeled flow by the trained ANNs.

4.4. Module 4: detection of global anomalies via PCA decomposition

PCA is a coordinate transformation method that has been previously used to detect traffic anomalies in networked systems (Huang et al. 2007; Lakhina et al. 2004; Lee et al. 2013). It can be used to re-map a given set of multi-dimensional data points onto new axes known as the principal components (PCs). Each PC points in the direction representing the maximum variance remaining in the data after accounting for the variance in the preceding components. Hence, the first PC captures the maximum variance in the data that can be projected on a single axis, while the following orthogonal PCs capture the remaining variance, with each component capturing the largest variance for the next orthogonal direction. The set of M principal components $\{\boldsymbol{p}_l\}_{l=1}^{M}$ is defined as (Lakhina et al. 2004):

$$\boldsymbol{p}_l = \arg\max_{\|\boldsymbol{v}\|=1} \left\| \left(\boldsymbol{Z} - \sum_{j=1}^{l-1} \boldsymbol{Z}\boldsymbol{p}_j\boldsymbol{p}_j^T \right) \boldsymbol{v} \right\| \tag{7}$$

which can be solved by evaluating the M eigenvectors of the covariance data matrix: $\boldsymbol{\Sigma_Z} = \frac{1}{N}\boldsymbol{Z}^T\boldsymbol{Z}$, where $\boldsymbol{Z}$ is the standardized (z-score) version of the training observations matrix $\boldsymbol{A}$. Evaluation of the principal components and projections of the sensory data are conducted using the Statistics and Machine Learning Toolbox™ incorporated in MATLAB R2016a.

4.4.1. Projections on the Normal and Anomalous Subspaces

In the context of anomaly detection, PCs can be split into two sets corresponding to the normal and anomalous subspaces (Lakhina et al. 2004; Ringberg et al. 2007). The first set representing the "normal subspace" consists of the PCs that contain most of the natural variation in the data. On the other hand, the rest of the PCs that only capture minimal variability are representative of the "anomalous subspace" in which the projections of anomalous observations appear more

distinctive. Herein, the first 14 principal components $\{\boldsymbol{p}_l\}_{l=1}^{14}$ are found to capture 99% of the variance in the training data matrix, which unveils the low intrinsic dimensionality of the observations matrix. Therefore, they are considered representative of the normal subspace, while the rest of the principal components $\{\boldsymbol{p}_l\}_{l=15}^{31}$ represent the anomalous subspace. To demonstrate the difference between the two subspaces, Figure 5 shows the projections of the validation dataset observations on PC1 and PC18 representing the normal and anomalous subspaces, respectively. For the projections on PC18, anomalous observations induced by the first four attacks clearly appear to have either excessively high or low values compared to other observations reported when the system is not under attack, which facilitates their detection. On the other hand, projections on PC1 comprise no clear outliers, even during an ongoing attack, since they mostly correspond to the normal periodic patterns of the data. In this study, anomalous observations are defined as the ones yielding projections that are further than γ standard deviations above or below the mean projection ($\mu_l \pm \gamma\sigma_l$) for one or more of the anomalous subspace principal components.

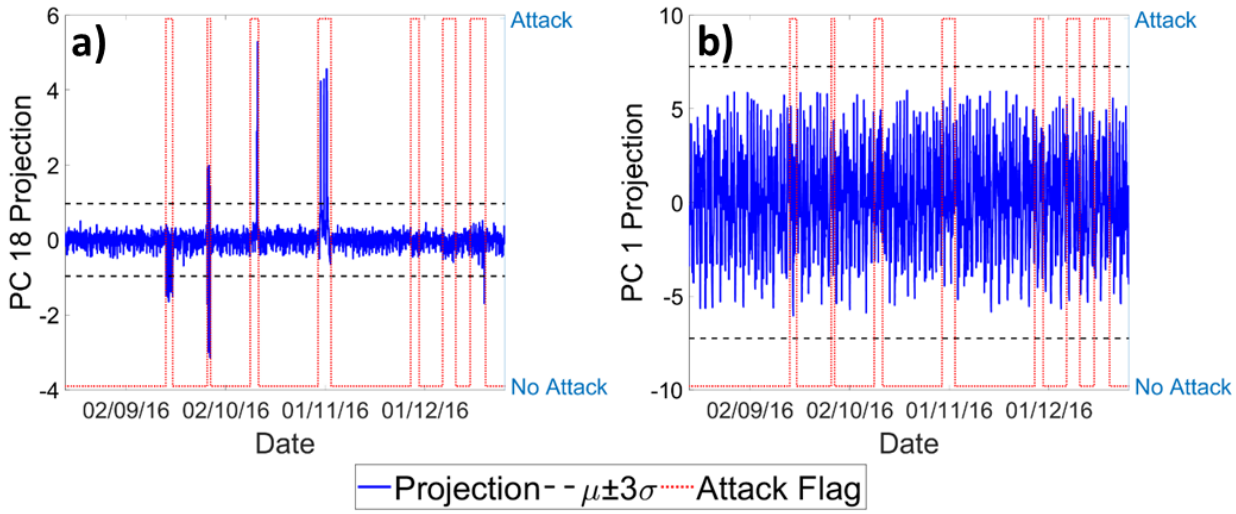

Figure 5. Projections of the validation dataset on: (a) PC18 (anomalous subspace), and (b) PC1 (normal subspace).

4.4.2. Leave One Out (LOO) Approach

In addition to directly using the projections of the data points on the PCs to find anomalies, we also implement the leave one out (LOO) methodology demonstrated by Lee et al. 2013, which examines the effect of adding the observations of interest on changing the direction of the principal components. We first start by evaluating the initial PCs of the training observations matrix $\boldsymbol{A}$. Then we proceed by adding each of the observations from the validation or test datasets to the training dataset, and then re-evaluating the principal components each time. With adding a new observation, the directions of the resulting PC vectors are expected to deviate from the original directions, and the angle of the deviation will be dependent on the outlier-ness of the added observation. Therefore, an anomalous data instance will yield a larger deviation than the

one generated by adding a normal data instance as shown in Figure 6-a. The deviation can be quantified by calculating the cosine similarity between the directions of the PCs before and after adding the observation as:

$$Sim_l = 1 - \cos(\theta) = 1 - \left| \frac{\langle p_l, \hat{p}_l \rangle}{\|p_l\| \|\hat{p}_l\|} \right| \quad (8)$$

where p_l and $\hat{p}_l$ are the principal component vectors before and after adding the data instance. According to this definition, the similarity can take any value between 0 and 1, with ($Sim = 1$) corresponding to the largest deviation (i.e. p_i and $\hat{p}_i$ are orthogonal), and ($Sim = 0$) representing the smallest deviation (i.e. p_i and $\hat{p}_i$ have similar orientation). Figure 6-b shows the similarity plot inflicted by the projections of the validation dataset on PC15 of the anomalous subspace. Anomalous points corresponding to attack events have clearly larger deviations relative to normal observations collected when the system is not under attack. Thus, observations having a cosine similarity more than γ standard deviations above the mean ($\mu + \gamma\sigma$) are considered anomalous.

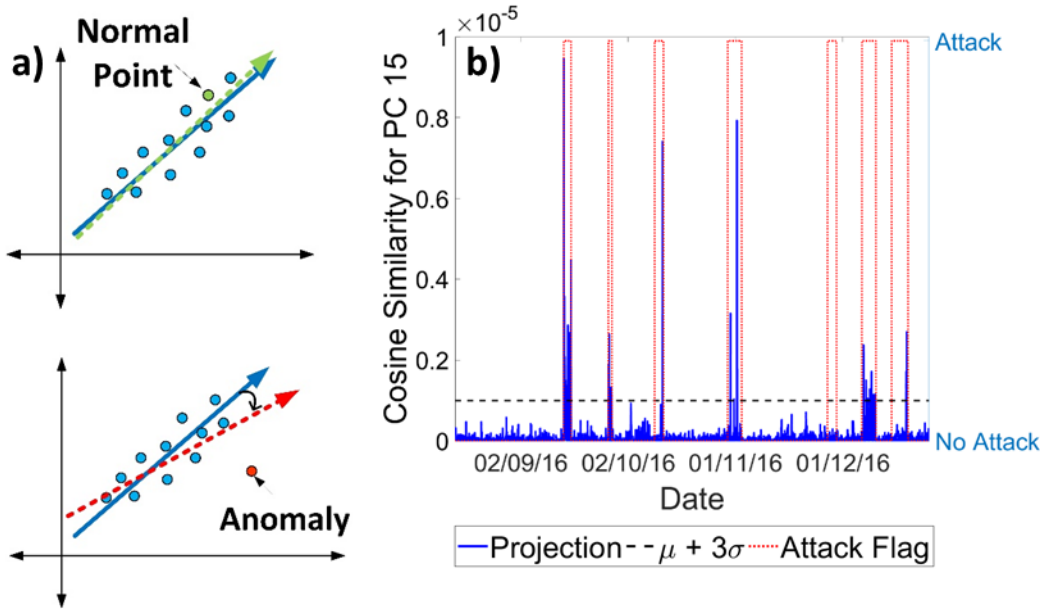

Figure 6. (a) Schematic of the Leave One Out (LOO) method for a normal vs anomalous observation, and (b) Cosine similarity for projections on PC 15.

4.5. Integration of the four modules and real-time operation

4.5.1. Alarm watch *window*

The detection algorithm operates in a real-time fashion, where the time-series observations are analyzed one-by-one by each of the four modules (Figure 2). If any of the four modules discovers an anomaly in the incoming data stream, the algorithm raises a flag to mark the anomalous point. Since the anomaly can be induced either by a benign data point or by an actual attack, the algorithm puts the system under watch for a user-specific *window* time-period. If a

second anomaly is detected during the watch *window*, the entire period between the first and second anomalies is flagged, and the watch is extended for another *window* starting from the second discovered anomaly. This process is repeated until no more anomalies are discovered within a *window* period starting from the last discovered anomaly. The algorithm issues an attack alarm only if the total flagged period between the first and last discovered anomalies is longer than a user-specific threshold period, which in this study is taken equal to the *window* period for simplicity. Conversely, if no alarm is issued, the algorithm stops the watch and discards the discovered flags. Benign observations are then added to the pool of trusted data that is used to re-train modules 2, 3 and 4.

While the employed approach performs well in detecting the attacks featured in the BATADAL dataset, enhancements can still be made to the way the algorithm implements the *window* period method. For example, the frequency and magnitude of anomalies discovered during the watch *window* can be factored in. This means that anomalies simultaneously detected by more than one module, or whose deviations from the detection thresholds are very large, should have more weight in determining the existence of an attack. Another modification can be by changing the detection thresholds to become more stringent during the watch *window*. The algorithm can also use a probability-based method similar to those employed by previous studies (Arad et al. 2013; Perelman et al. 2012), which can be done by recursively updating the probability of an event with the detection of every new anomaly, using Bayes' rule for example, to decide whether an anomalous sequence of observations corresponds to an event.

4.5.2. Module retraining and thresholds update

To elucidate the re-training processes of the three modules, assume we start with the previously mentioned training observations matrix $\boldsymbol{A}(N \times M)$. The incoming vector of observations from all monitors at the following time-step $N+1$ can be written as $[x_{N+1,1}, \dots, x_{N+1,i}, \dots, x_{N+1,M}]$. For instance, this can correspond to the observations made by all M monitors during the first time-step in the validation or test datasets. If no anomalies are discovered, this vector is added to the training data matrix $\boldsymbol{A}$, thus increasing its size by one row, i.e. $([N+1] \times M)$. The user can specify the maximum permissible size for the training data matrix, beyond which, the oldest observations will be discarded to accommodate newer ones. Module 2 is re-trained by re-calculating all the previously defined statistical properties (μ_i, σ_i, $Q1_i$, and $Q3_i$) and thresholds ($n_i^{u,l}$, and $m_i^{u,l}$). For module 3, ANN models for all monitors are re-trained, and the maximum training error (ϵ_i^{train}) is re-evaluated for each monitor to update its detection threshold. Module 4 is re-trained by re-evaluating the set of M principal components $\{\boldsymbol{p}_l\}_{l=1}^{M}$, and re-calculating the mean and standard deviation of the projections (μ_l, and σ_l) on the anomalous subspace PCs to update their detection thresholds. It should be noted that, while the threshold coefficients for modules 3 and 4, i.e. α and γ, are fixed throughout the process, the detection thresholds are updated during retraining since they depend on ϵ_i^{train}, μ_l, and σ_l. These dynamic thresholds are particularly useful for systems with large seasonal variations in water demands, which was not the case in the BATADAL datasets.

The frequency at which each of the three modules is re-trained using the updated data matrix is user-specific, but is also subject to some restrictions. While modules 2 and 4 can be retrained with every new row of observations added to the data matrix, re-training the ANN models is restricted by having a sequence of $I + 1$ benign observations with no interrupting attacks. In addition, retraining the ANN models is the most computationally demanding among the three modules, which may limit how frequently they can be re-trained without needing additional computational resources. Algorithm users can set the initial weight coefficients for the re-training process to those of the previously trained ANN models instead of random values to reduce the computational burden. Besides, while not examined herein, users may want to periodically discard, or at least double-check, some of the anomalous observations that do not end up being labeled as attacks instead of using them for retraining. Accordingly, anomalous instances that are extremely far from the alarm thresholds, or that raise flags by more than one module are good candidates for exclusion.

5. Results and Discussion

5.1. Algorithm validation and performance tuning

To assess the efficiency of the proposed technique in identifying CPAs, the trained algorithm is applied to the second (validation) dataset with seven simulated attacks. A combined performance metric score is implemented to quantify the efficiency of the algorithm in detecting attacks both quickly and reliably and to tune its performance by finding the best *window*.

5.1.1. Performance Metrics

Using a similar approach to the one adopted in the BATADAL (Taormina et al. 2018), algorithm performance is evaluated using a combined score metric (S) that consists of two components, namely Time-To-Detection (S_{TTD}) which reflects the time required for detecting a threat, and Confusion-Matrix (S_{CM}) which quantitatively determines the quality of recognizing true threats:

$$S = (S_{TTD} + S_{CM})/2 \tag{9}$$

The first component S_{TTD} represents the performance metric for the time to detection as a ratio of the total attack duration ΔT for all the existing attacks in the dataset:

$$S_{TTD} = 1 - \frac{1}{N_A}\sum_{i=1}^{N_A}\frac{TTD_i}{\Delta T_i} \tag{10}$$

where N_A is the number of attacks. According to this definition, S_{TTD} can take any value between zero and 1, with $S_{TTD} = 0$ representing the worst case scenario where none of the attacks is detected while they are taking place (i.e. $TTD_i = \Delta T_i$), while $S_{TTD} = 1$ represents the ideal case scenario where all attacks are detected immediately (i.e. $TTD_i = 0$). The second component S_{CM} represents the Area Under the Curve (AUC) which is a metric used to maximize true alarms while minimizing false alarms (Powers 2011):

$$S_{CM} = \frac{TPR + TNR}{2} \tag{11}$$

TPR is the Ture-Positive-Rate, which is the ratio of true positive (TP) alarms to the sum of true-positives (TP) and false-negatives (FN), and thus it represents the sensitivity of the algorithm to detect true threats. TNR is the True-Negative-Rate, which is the ratio of true negative (TN) alarms to the sum of true-negatives and false-positives (FP), and hence TNR quantifies the specificity of the algorithm in recognizing when the system is not under attack. Therefore, S_{CM} can take any value between 0 and 1, with $S_{CM} = 1$ representing and ideal case where no false positives and no false negatives are labeled by the algorithm, while $S_{CM} = 0$, represents the worst case where the algorithm can't label any true positives or negatives. Nevertheless, S_{CM} should practically be greater than 0.5, which is the score achieved by a naïve algorithm that predicts that the system is under attack at all times. Hence, the overall performance score should take a value in the range [0.75-1] for the detection technique to be deemed successful.

5.1.2. Attack detection performance

Figure 7-a shows a plot of the attacks detected by the algorithm compared to the actual attacks in the validation dataset. As can be seen from the figure, the algorithm is able to detect all the seven simulated attacks, together with one false attack of a relatively shorter duration (between the fourth and fifth attacks). For all the detected attacks, the algorithm labels the start of the threat almost at, or even slightly before, the official attack start-time. This means that the time to detection (the time needed by the algorithm to recognize an attack) is near zero for all the labeled attacks ($\forall\, i, TTD_i \cong 0$), which is translated to a high *TTD* score of ($S_{TTD} = 0.984$). This can be attributed to the fact that an alarm watch is immediately triggered at the detection of the first anomaly, which may happen to be induced by a benign observation right before the beginning of the attack. This is followed by the discovery of other anomalies during the *window* period, which are combined by the algorithm and are eventually translated into a labeled attack. Additionally, Figure 7-a shows that the algorithm is able to maintain the true alarm status for as long as the actual attack is still active. This is mainly because the algorithm extends the watch period until no extra flags are discovered within a similar *window* starting from the last discovered flag. However, this also leads to the algorithm falsely extending the alarm periods for a few time steps after the formal end of the actual attack for most attacks (attacks 1, 3, 5 and 6). It is noteworthy that the time-to-detection (TTD) as defined in the BATADAL does not reflect the practical time taken by the proposed algorithm when operated in a real-time fashion. This is because the attack alarm is issued after a *window* period has elapsed starting from when the first anomaly is discovered. Nevertheless, since the algorithm was able to label all the simulated attacks at or before their actual start, the practical TTD for real-time operation is expected to be less than the *window* period.

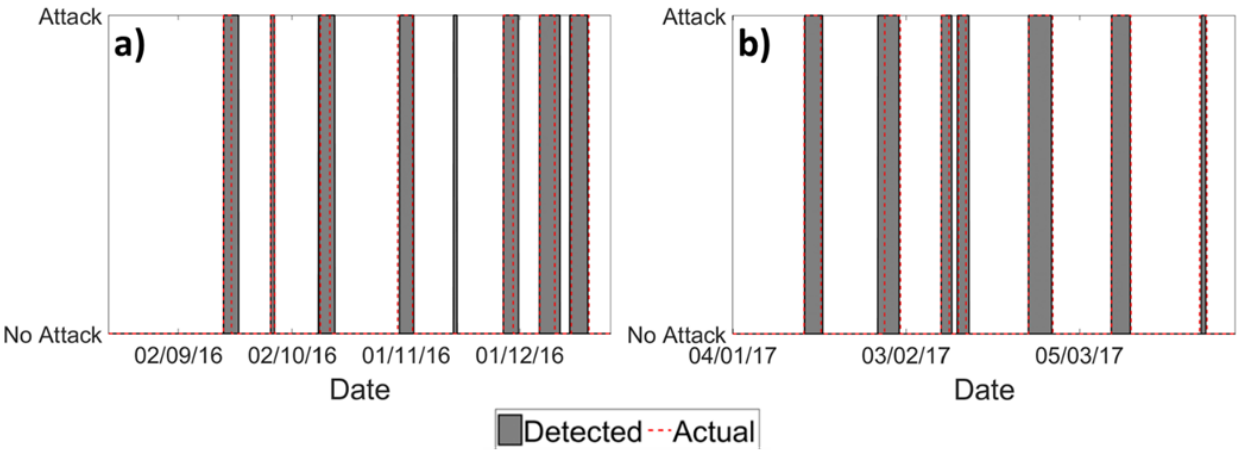

Figure 7. Detected vs. labeled attack for: (a) validation dataset, and (b) test dataset.

For the validation dataset, the confusion matrix score of the algorithm is slightly less than the time to detection score ($S_{CM} = 0.953$), which is attributed to its low true negative rate ($TNR = 0.946$). This is due to the relatively large number of false positives ($FP = 4.764\%$) that mainly correspond to the extended alarm periods after the end of the true attacks as well as the one false attack detected. Nevertheless, the algorithm has a significantly lower number of false negatives ($FN = 0.479\%$), which is more essential since undetected attacks can yield serious consequences if they resulted in physical damage to the infrastructure or if the water quality is compromised. The overall performance score of the algorithm for the validation dataset is sufficiently close to unity ($S = 0.968$), indicating a satisfactory performance.

5.1.3. Role of each module

Figure 8 shows the attacks detected by each of the four modules for the validation dataset. It can be seen that, for most of the detected attacks, each module was able to pick up some of the anomalies. Nevertheless, none of the modules was able to detect all of the seven simulated attacks independently. Thus, compiling the flags detected by all four modules is necessary to detect all attacks with high speed and reliability. The third and fourth modules (ANNs, and PCA) appear to be the most efficient in terms of accurately detecting most of the attacks at the right start times, while each of the first and second modules fully detected only two of the seven attacks. Moreover, the drop in the calculated performance score when only the third and fourth modules are used to detect attacks is negligible ($S = 0.966$), reflecting the dominance of the ANN and PCA methods.

It is worth mentioning that the fourth module (PCA) is the only one that appropriately detected the seventh attack event, while the three other modules failed to detect it even partially. During this attack, the individual behavior of each of the attacked network elements does not display any exceptionally suspicious inconsistencies due to attack concealment. Nevertheless, projecting their combined data on the principal components of the anomalous subspace shows a clear anomalous sequence during the seventh attack event (Figure S4). PCA was able to detect

this attack since it takes into consideration the naturally existing correlation between the collected readings from different sensors in order to identify anomalous behaviors that can go undetected when data from each sensor is only analyzed independently. While this demonstrates the critical role of the fourth module in detecting such global anomalies, it is important to note that the PCA module also yields a group of sporadic falsely labeled flags induced by benign observations. However, most of these flags are eventually discarded by the algorithm because they are not typically followed by any more flags within the *window* period.

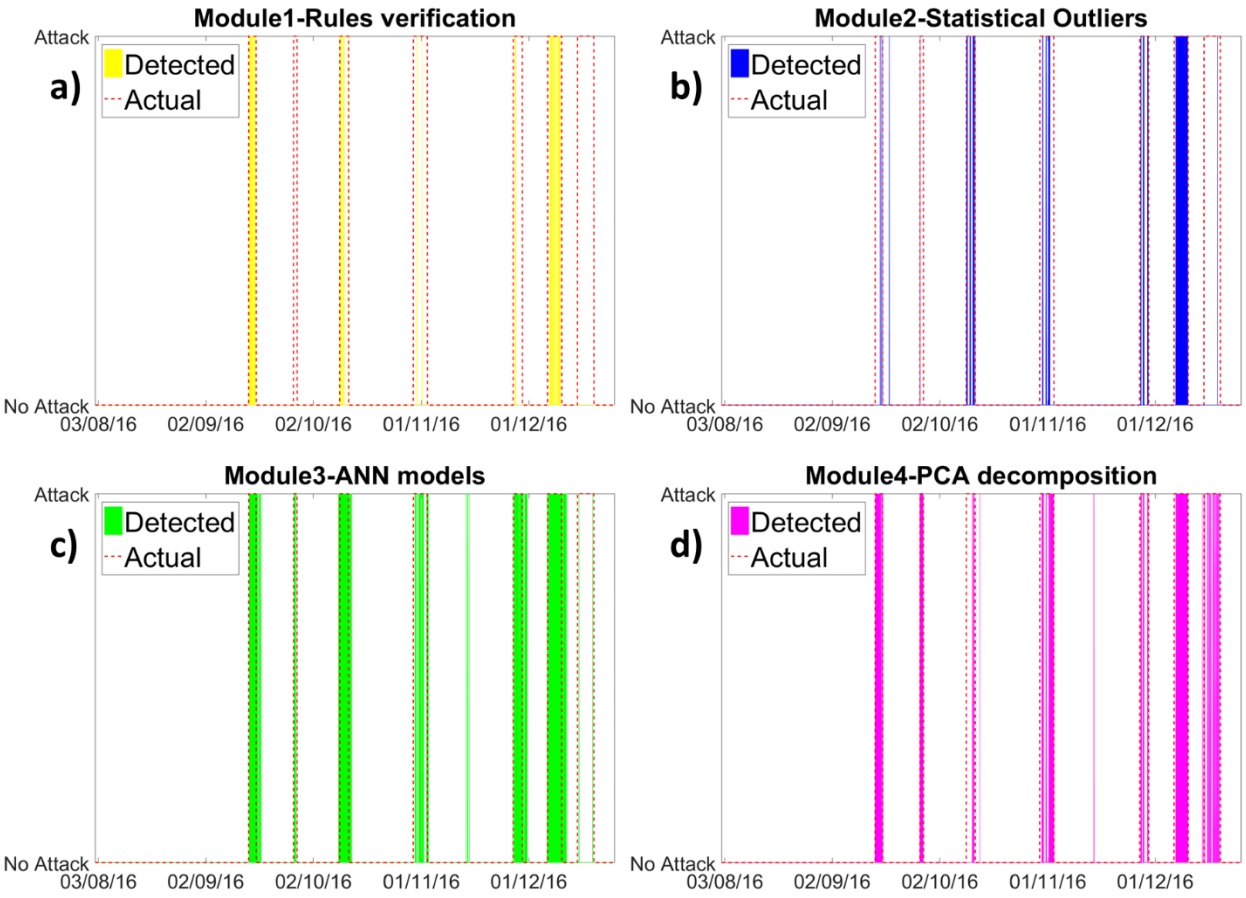

Figure 8. Role of each module in attacks detection for the validation dataset

5.1.4. Optimal alarm *window*

The fact that the algorithm flags some of the benign observations sheds light on two important aspects of the proposed approach: (i) the sensitivity of the detected outliers to the threshold limits, and (ii) the effect of the *window* size on the algorithm performance. It is imperative that using a more stringent threshold criteria for defining an anomalous observation would typically improve the detection of attacks by minimizing the number of false negative alarms, yet, this would also lead to a higher number of false-positive alarms because noisy observations that do not necessarily correspond to an attack can still be flagged. Therefore, before applying the algorithm to new datasets, calibration should be conducted to draw the appropriate line between

normal and anomalous data instances based on the characteristics of the system. While the second module requires no calibration since the detection thresholds ($n_i^{u,l}$, and $m_i^{u,l}$) are defined based on the maximum and minimum values in the training dataset, the threshold coefficients for the third and fourth modules, i.e. α and γ, require calibration. In this study, these parameters were selected by enumeration to yield the best performance for each module separately ($\alpha = 5$, and $\gamma = 3$). Another alternative to this approach is to use a different threshold coefficient for the ANN model of each monitor $\{\alpha_i\}_{i=1}^{M}$, as well as each principal component $\{\gamma_i\}_{i=1}^{M}$ instead of the global settings used herein. While this will increase the number of calibration parameters from two to $2M$, it is expected to enhance the performance of the algorithm. An optimization routing, e.g. genetic algorithm, can be used to perform the calibration instead.

A similar effect can also be inflicted by the chosen *window*, since a larger *window* means that the system is put under an alarm watch for extended periods of time even when no flags are being detected. On the other hand, a shorter *window* can result in missing most of the attacks since the majority of the flags will be discarded within a short period if no more flags are discovered. Figure S5-a shows the effect of the size of the alarm-watch *window* period on each of the three performance metrics. It demonstrates that an optimal *window* period can be selected to achieve the best performance, while smaller and larger windows would generally yield less accurate results. It is noteworthy here that finer sampling frequencies can enable the users to choose a shorter *window* without compromising on the accuracy. This will shorten the delay period before an attack alarm is issued after a series of anomalous observations is detected.

5.1.5. Attack localization

Only the first three modules of the detection algorithm are capable of attack localization, which is identifying the actual components of the cyber-physical system that were directly targeted or affected by the attack. This is because they rely on analyzing the data from each sensor/actuator individually, while the fourth module (PCA) only recognizes global anomalies without specifying which component is the actual source of the anomaly. For all the detected attacks, the algorithm picks anomalies in the observations obtained from multiple sensors/actuators at the same time; however, most of these anomalies are typically concentrated in the data from only two or three elements. Table 1 lists the localized attacks as identified by the algorithm from the anomalies in the validation dataset, while Table S2 shows the actual attacked components during each attack scenario. For instance, during the first attack, the water level information of Tank 7 is manipulated, which jeopardizes the operations of Pumps 10 and 11. While the algorithm does not specify Tank 7 as an attacked component, it still discovers the inflicted anomalous observations in the flow of the corresponding pumps. Thus, for this attack, the algorithm can only give multiple probable scenarios without necessarily specifying which one is true. For example, the detected anomalies can correspond to a direct attack on the actuators connected to the pumps, the manipulation of the connection between Tank 7 sensor (PLC9) and pumps actuators (PLC5), or a direct attack on the tank sensor (PLC9). Similarly, for most of the attacks, the algorithm is able to identify either the actual targeted components or the sensors/actuators that immediately

control or are controlled by its operation. However, this is not the case for the fourth attack, where the readings of Tank 1 level sent to PLC2 are manipulated and the operation of pumps 1 and 2 is compromised, yet the corresponding anomalies are picked up from multiple elements. The majority of these anomalies are concentrated in the suction pressure observations of Valve 2 and pumping stations 4 and 5, which are distributed across more than one PLC, and do not have any common connectivity with the truly targeted components. Thus, for this attack, the algorithm is neither able to localize the actual attacked components, nor the immediately affected ones. Still, such confusion is expected based on the study of Taormina et al. (2017), where they showed that the same hydraulic response for the WDS can be reached due to drastically different attack scenarios.

Table 1. Attack localization results for the validation and test datasets

Dataset	Att#	Detected Attack Components
Validation	1	PU11 Flow / PU10 Flow / PS5 DP
	2	PS5 DP
	3	T1 Level / PU1 Flow / PS1 DP / V2 SP
	4	V2 SP / PS5 SP / PS4 SP
	5	PU6 Flow / T4 Level
	6	PU6 Flow / PU7 Flow
	7	PS3 DP
Test	1	PS2 DP
	2	V2 DP / PS2 SP / PS3 SP / PU7 Flow / T2 Level
	3	PU3 Flow / PU2 Flow
	4	PU3 Flow / PU1 Flow / T1 Level / PU2 Flow
	5	PS2 SP / PS3 SP / PU7 Flow
	6	PS4 SP / T6 Level / V2 SP
	7	PS3 DP

5.2. Algorithm testing and effect of measurement noise

5.2.1. Detecting Attacks in the test dataset

The tuned algorithm is finally tested against the third test dataset containing approximately 3 months of observations and featuring seven different attack scenarios. This fresh dataset has not been involved in either the training or the validation of the algorithm and thus can reflect the actual performance in detecting CPAs in new observations. Figure 7-b displays the actual attacks featured in the dataset plotted against the attacks detected by the algorithm. Similar to the validation dataset, the algorithm detects all the seven attacks in the test dataset demonstrating high detection efficiency. In addition, the algorithm identifies the existence of the threat at or before the formal start time of the actual attack for most of the attacks. Nevertheless, the time-to-detection score for the test dataset ($S_{TTD} = 0.963$) is clearly less than the validation dataset. This can be attributed to the delay encountered in detecting the seventh attack, which also happened to be the shortest attack in the test dataset resulting in a high ($TTD_i/\Delta T_i$) ratio. Although the percentage of false positives in the test dataset ($FP = 3.159\%$) is less than that encountered in

the validation dataset, the confusion matrix score for the test dataset ($S_{CM} = 0.947$) is still lower than the validation dataset. This is mainly because the test dataset yields a higher percentage of true negatives ($FN = 1.292\%$) than the validation dataset due to the delayed identification of the seventh attack scenario. Thus, the overall performance score of the test dataset (S=0.955) is lower than that of the validation dataset, which is expected since the selection of the *window* period was conducted for the latter.

Figure S6 displays the labeled attacks by each of the four modules for the test dataset. The fourth module (PCA decomposition) is the most efficient in appropriately detecting all the seven attacks. Similar to the validation dataset, PCA was the only module capable of detecting the sixth attack that induces only minor anomalies in the observations of each individual sensor/actuator due to concealment, and thus can only be discovered by a coordinate transformation method. It is worth noting that the first module does not detect any attacks in the test dataset since during the attacks all the actuator rules were precisely followed. This scenario can take place if the attacker had intimate knowledge of the system, which could be gained through eavesdropping on network communications. This allows the attacker to conceal the apparent sensor/actuator readings to obey the rules at all times, corresponding to a highly sophisticated attack. Table 1 lists the localized attacks as identified by the algorithm from the anomalous observations in the test dataset. Similar to the validation dataset, the algorithm identifies the actual compromised or the immediately affected network components for the majority of the attacks.

5.2.2. Effect of noise

Since the algorithm adopts a "data-based" approach for the detection of cyber-physical attacks by isolating anomalous observations, it is important to test its robustness against outliers caused by measurement noise that does not necessarily correspond to an attack. So far, the datasets that were used in the training, validation, and testing of the algorithm are considered to correspond to a group of perfect monitors. However, a real-life sensor signal will typically constitute some degree of undesired noise that either takes the form of a continuous or a semi-continuous uniform background noise, or can be a group of outliers with different magnitudes that are randomly encountered across the time series signal. The algorithm can readily isolate and discard the flags inflicted by the latter form of noise thanks to the *window* period rule as previously demonstrated. Nevertheless, the former type of uniform noise can be confusing to the detection algorithm because of its continuous nature, which might cause a series of anomalous observations that can be interpreted by the algorithm as an attack.

To test the robustness of the detection technique to sensor noise, the algorithm is applied to the test dataset after adding Gaussian background noise to the observations in the test dataset. For each observation, a noise component is randomly generated from a normal distribution with a zero mean and a standard deviation equivalent to a chosen fraction of the standard deviation of the clear signal ($\sigma_{noise}/\sigma_{signal}$) and then added to the clear signal. A range of different standard deviations is tested, starting from zero up to 0.5, which correspond to a perfectly clear signal and

a highly noisy signal, respectively. Figure S5-b shows the three performance scores for each tested scenario with different ($\sigma_{noise}/\sigma_{signal}$) ratios. As expected, the detection performance of the algorithm drops as the signal noise increases. The overall performance score (S) drops consistently due to the decrease of the confusion matrix score (S_{CM}), while the time to detection score does not change much. It should be noted that for these tests, the second module (simple outlier detection) is turned off. Figure S7 demonstrates that the extended periods of anomalous observations caused by the signal noise lead the algorithm to label them as attacks, which resulted in a near trivial solution (S=0.761) for the case of $\sigma_{noise}/\sigma_{signal} = 0.5$, where the algorithm labels almost the entire simulation period as a continuous attack event. Thus, noise reduction might be crucial for successfully applying this approach if the data has a high degree of noise.

6. Conclusions

Implementing smart networking technologies in the sector of water infrastructure systems has expanded the domain of potential threats from the traditional risks associated with direct physical attacks that commonly aim at sabotaging the infrastructure equipment or compromising the water quality, to the risks of malicious attacks originating in the cyber-space. In this study, an algorithm is developed for the detection of cyber-physical attacks on smart water distribution systems. The algorithm employs multiple anomaly detection techniques to spot different types of anomalous observations in the sensory data. Artificial neural network models are first trained to predict the regular patterns of the system's operation and then used to identify suspicious observations that are inconsistent with the normal behavior of the system. Principal component analysis is conducted to decompose the multi-dimensional sensory data matrix into two sub-spaces representing the normal and the anomalous projections, which efficiently discovers global anomalies that are caused by highly concealed attacks.

The algorithm performs well in detecting all the simulated attacks in the validation and test datasets with short or no delays. Nevertheless, it tends to put the system under a false attack status for a few time-steps after the real threat no longer exists. Integrating the flags detected by all modules is shown to be necessary to detect and localize all attack scenarios with high efficiency. Yet, the principal component analysis appears to be the most reliable component of the algorithm, especially in discovering stealthy attacks that induce minor inconsistencies in the sensory data because of attack concealment. For most of the identified attacks, the algorithm was generally able to localize the system elements that are impacted by the threat. Nevertheless, for one of the attacks, the algorithm could identify neither the actual targeted elements, nor those that are immediately affected by their operation, because the discovered anomalies were dispersed in the observations from multiple unrelated sensors/actuators.

The algorithm shows a robust performance in the face of mild measurement noise. However, high noise adversely affects the detection ability as the algorithm confuses the extended anomalies caused by random noise with those inflicted by an actual attack. In addition, several anomalous events can still take place during usual operation of systems, such as pipe

bursts or pump shut-offs, which might be interpreted by the algorithm as a deliberate attack on the system.

Acknowledgments

Partial support from the Lucy and Stanley Lopata Endowment at Washington University in St. Louis is gratefully acknowledged.

References

Abokifa, A. A., Haddad, K., Lo, C. S., and Biswas, P. (2017). "Detection of Cyber Physical Attacks on Water Distribution Systems via Principal Component Analysis and Artificial Neural Networks." *Proceedings of World Environmental and Water Resources Congress 2017*, 676–691.

Abokifa, A. A., Yang, Y. J., Lo, C. S., and Biswas, P. (2016). "Water quality modeling in the dead end sections of drinking water distribution networks." *Water Research*, 89, 107–117.

Almalawi, A., Fahad, A., Tari, Z., Alamri, A., Alghamdi, R., and Zomaya, A. Y. (2016). "An Efficient Data-Driven Clustering Technique to Detect Attacks in SCADA Systems." *IEEE Transactions on Information Forensics and Security*, 11(5), 893–906.

Amin, S., Litrico, X., Sastry, S., and Bayen, A. M. (2013a). "Cyber security of water SCADA systems - Part I: Analysis and experimentation of stealthy deception attacks." *IEEE Transactions on Control Systems Technology*, 21(5), 1963–1970.

Amin, S. M., Litrico, X., Sastry, S. S., and Bayen, A. M. (2013b). "Cyber Security of Water SCADA Systems - Part II: Attack Detection Using Enhanced Hydrodynamic Models." *IEEE Transactions on Control Systems Technology*, 21(5), 1679–1693.

Arad, J., Housh, M., Perelman, L., and Ostfeld, A. (2013). "A dynamic thresholds scheme for contaminant event detection in water distribution systems." *Water Research*, 47(5), 1899–1908.

Broad, D. R., Dandy, G. C., and Maier, H. R. (2015). "A systematic approach to determining metamodel scope for risk- based optimization and its application to water distribution system design." *Environmental Modeling & Software*, 69, 382–395.

Broad, D. R., Maier, H. R., and Dandy, G. C. (2010). "Optimal Operation of Complex Water Distribution Systems Using Metamodels." *Journal of Water Resources Planning and Management*, 136(4), 433–443.

Chandola, V., Banerjee, A., and Kumar, V. (2009). "Anomaly Detection: A Survey." *ACM computing surveys (CSUR)*, 41(3), 15–58.

Cunningham, J. P., and Ghahramani, Z. (2014). "Linear Dimensionality Reduction: Survey, Insights, and Generalizations." *Journal of Machine Learning Research*, 16, 2859–2900.

DHS ICS-CERT. (2015). "U.S. Department of Homeland Security – Industrial Control Systems-

Cyber Emergency Response Team, Year in Review." *https://ics-cert.us-cert.gov/*.

Eliades, D. G., and Polycarpou, M. M. (2012). "Leakage fault detection in district metered areas of water distribution systems." *Journal of Hydroinformatics*, 14(4), 992–1005.

Gao, W., Morris, T., Reaves, B., and Richey, D. (2010). "On SCADA control system command and response injection and intrusion detection." *eCrime Researchers Summit, IEEE 2010*.

Hawkins, D. M. (1980). *Identification of outliers*. Springer.

Horta, R. (2007). "The city of Boca Raton: A case study in water utility cybersecurity." *Journal - American Water Works Association*, 99(3), 48–55.

Housh, M., and Ohar, Z. (2017a). "Integrating physically based simulators with Event Detection Systems: Multi-site detection approach." *Water Research*, 110, 180–191.

Housh, M., and Ohar, Z. (2017b). "Multiobjective Calibration of Event-Detection Systems." *Journal of Water Resources Planning and Management*, 143(8), 06017004.

Housh, M., and Ohar, Z. (2018). "Model-based Approach for Cyber-Physical Attack Detection in Water Distribution Systems." *Water Research*, 139, 132–143.

Huang, L., Nguyen, X., Garofalakis, M., Jordan, M. I., Joseph, A., and Taft, N. (2007). "In-Network PCA and Anomaly Detection." *Advances in Neural Information Processing Systems*, 19, 617–624.

Izquierdo, J., López, P. A., Martínez, F. J., and Pérez, R. (2007). "Fault detection in water supply systems using hybrid (theory and data-driven) modelling." *Mathematical and Computer Modelling*, 46(3–4), 341–350.

Jolliffe, I. T., and Cadima, J. (2016). "Principal component analysis : a review and recent developments." *Philosophical Transactions of the Royal Society A*, 374(2065), 20150202.

Kosut, O., Jia, L. J. L., Thomas, R. J., and Tong, L. T. L. (2010). "Malicious Data Attacks on Smart Grid State Estimation: Attack Strategies and Countermeasures." *Smart Grid Communications, IEEE 2010*, 220–225.

Lakhina, A., Crovella, M., and Diot, C. (2004). "Diagnosing network-wide traffic anomalies." *ACM SIGCOMM Computer Communication Review*, 34(4), 219–230.

Laszka, A., Abbas, W., Vorobeychik, Y., and Koutsoukos, X. (2017). "Synergic Security for Smart Water Networks: Redundancy, Diversity, and Hardening." *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*, 21–24.

Lee, E. A. (2008). "Cyber Physical Systems: Design Challenges." *11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing*, 363–369.

Lee, Y. J., Yeh, Y. R., and Wang, Y. C. F. (2013). "Anomaly detection via online oversampling principal component analysis." *IEEE Transactions on Knowledge and Data Engineering*, 25(7), 1460–1470.

Maglaras, L. A., and Jiang, J. (2014). “Intrusion detection in SCADA systems using machine learning techniques.” *Science and Information Conference, IEEE 2014*, 626–631.

Maier, H. R., and Dandy, G. C. (2000). “Neural networks for the prediction and forecasting of water resources variables: A review of modelling issues and applications.” *Environmental Modelling & Software*, 15(1), 101–124.

Mathur, A. (2017). “SecWater: A Multi-Layer Security Framework forWater Treatment Plants.” *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*, 29–32.

May, R. J., Dandy, G. C., Maier, H. R., and Nixon, J. B. (2008). “Application of partial mutual information variable selection to ANN forecasting of water quality in water distribution systems.” *Environmental Modelling & Software*, 23(10–11), 1289–1299.

Ohar, Z., Lahav, O., and Ostfeld, A. (2015). “Optimal sensor placement for detecting organophosphate intrusions into water distribution systems.” *Water Research*, 73, 193–203.

Panguluri, S., Nelson, T. D., and Wyman, R. P. (2017). “Creating a Cyber Security Culture for Your Water/Waste Water Utility.” *Cyber-Physical Security*, Springer International Publishing, 133–159.

Pasqualetti, F., Dorfler, F., and Bullo, F. (2013). “Attack detection and identification in cyber-physical systems.” *IEEE Transactions on Automatic Control*, 58(11), 2715–2729.

Perelman, L., Arad, J., Housh, M., and Ostfeld, A. (2012). “Event detection in water distribution systems from multivariate water quality time series.” *Environmental science & technology*, 46(15), 8212–8219.

Powers, D. (2011). “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.” *Journal of Machine Learning Technologies*, 2(1), 37–63.

Rasekh, A., Hassanzadeh, A., Mulchandani, S., Modi, S., and Banks, M. K. (2016). “Smart Water Networks and Cyber Security.” *Journal of Water Resources Planning and Management*, 142(7), 01816004.

Razavi, S., Tolson, B. A., and Burn, D. H. (2012). “Numerical assessment of metamodelling strategies in computationally intensive optimization.” *Environmental Modelling & Software*, 34, 67–86.

Ringberg, H., Soule, A., Rexford, J., and Diot, C. (2007). “Sensitivity of PCA for traffic anomaly detection.” *ACM SIGMETRICS Performance Evaluation Review*, 35(1), 109–120.

Romano, M., and Kapelan, Z. (2014). “Adaptive water demand forecasting for near real-time management of smart water distribution systems.” *Environmental Modelling & Software*, Elsevier, 60, 265–276.

Shamir, U., and Salomons, E. (2008). “Optimal Real-Time Operation of Urban Water Distribution Systems Using Reduced Models.” *Journal of Water Resources Planning and*

Management, 134(2), 181–185.

Slay, J., and Miller, M. (2008). "Lessons learned from the Marchoory Water Breach." *Critical Infrastructure Protection*, Boston: Springer, 73–82.

Sridhar, S., and Govindarasu, M. (2014). "Model-based attack detection and mitigation for automatic generation control." *IEEE Transactions on Smart Grid*, 5(2), 580–591.

Srirangarajan, S., Allen, M., Preis, A., Iqbal, M., Lim, H. B., and Whittle, A. J. (2013). "Wavelet-based burst event detection and localization in water distribution systems." *Journal of Signal Processing Systems*, 72(1), 1–16.

Taormina, R., Galelli, S., Tippenhauer, N. O., Salomons, E., and Ostfeld, A. (2016). "Simulation of cyber-physical attacks on water distribution systems with EPANET." *Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016*, 14, 123–130.

Taormina, R., Galelli, S., Tippenhauer, N. O., Salomons, E., and Ostfeld, A. (2017). "Characterizing cyber-physical attacks on water distribution systems." *Journal of Water Resources Planning and Management*, 143(5), 04017009.

Taormina, R., Galelli, S., Tippenhauer, N. O., Salomons, E., Ostfeld, A., Eliades, D. G., Aghashahi, M., Sundararajan, R., Pourahmadi, M., Banks, M. K., Brentan, B. M., Campbell, E., Lima, G., Manzi, D., Ayala-Cabrera, D., Herrera, M., Montalvo, I., Izquierdo, J., Luvizotto, E., Chandy, S. E., Rasekh, A., Barker, Z. A., Campbell, B., Shafiee, M. E., Giacomoni, M., Gatsis, N., Taha, A., Abokifa, A. A., Haddad, K., Lo, C. S., Biswas, P., Pasha, M. F. K., Kc, B., Somasundaram, S. L., Housh, M., and Ohar, Z. (2018). "The Battle of the Attack Detection Algorithms: Disclosing Cyber Attacks on Water Distribution Networks." *Journal of Water Resources Planning and Management*, 144(8), 04018048.